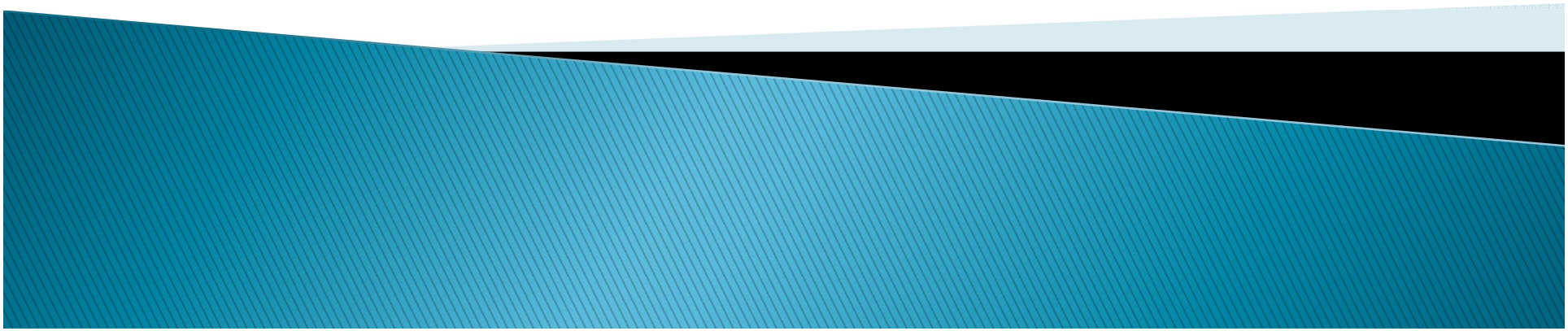
GOVERNMENT BETA
The Value of Unscripted Testing

Jonathan Haulund
Chief Software Engineer
United States Air Force

Rocky Khullar
Project Engineer
The Aerospace Corporation

Greg Dawson
Associate Vice President
LinQuest Corporation



Overview

- Introduction
- System Architecture
- Approach and Challenges
- Unscripted Test
- Day-in-the-Life Test
- Incremental Test
- Risk Management: Quality vs. Progress
- The Contract and Award Fee Plan
- Final Thoughts

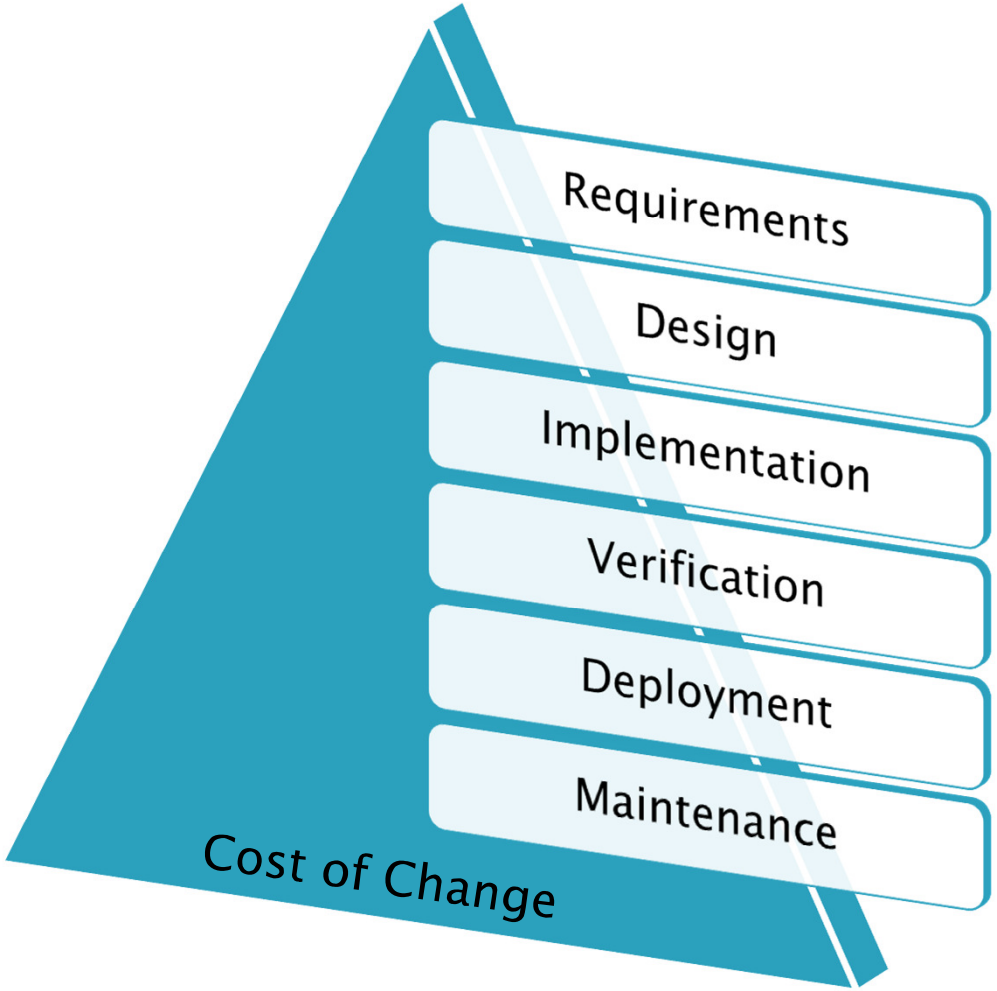
Introduction

- “Government Beta”

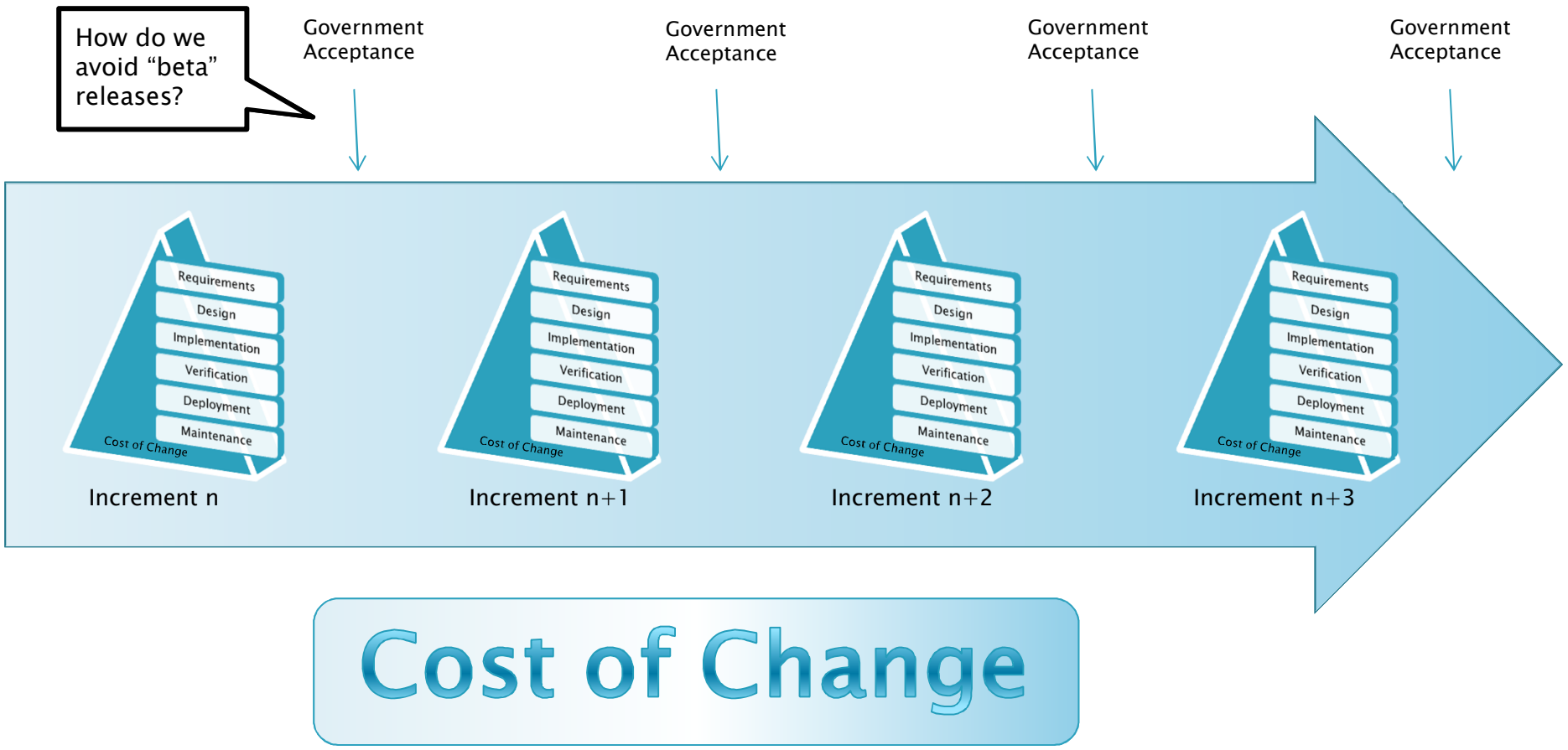
The Beta Effect

- If you want to see your software do what it was designed to do, put it in the hands of its users (literally and figuratively)

The Process of Development



The Incremental Effect

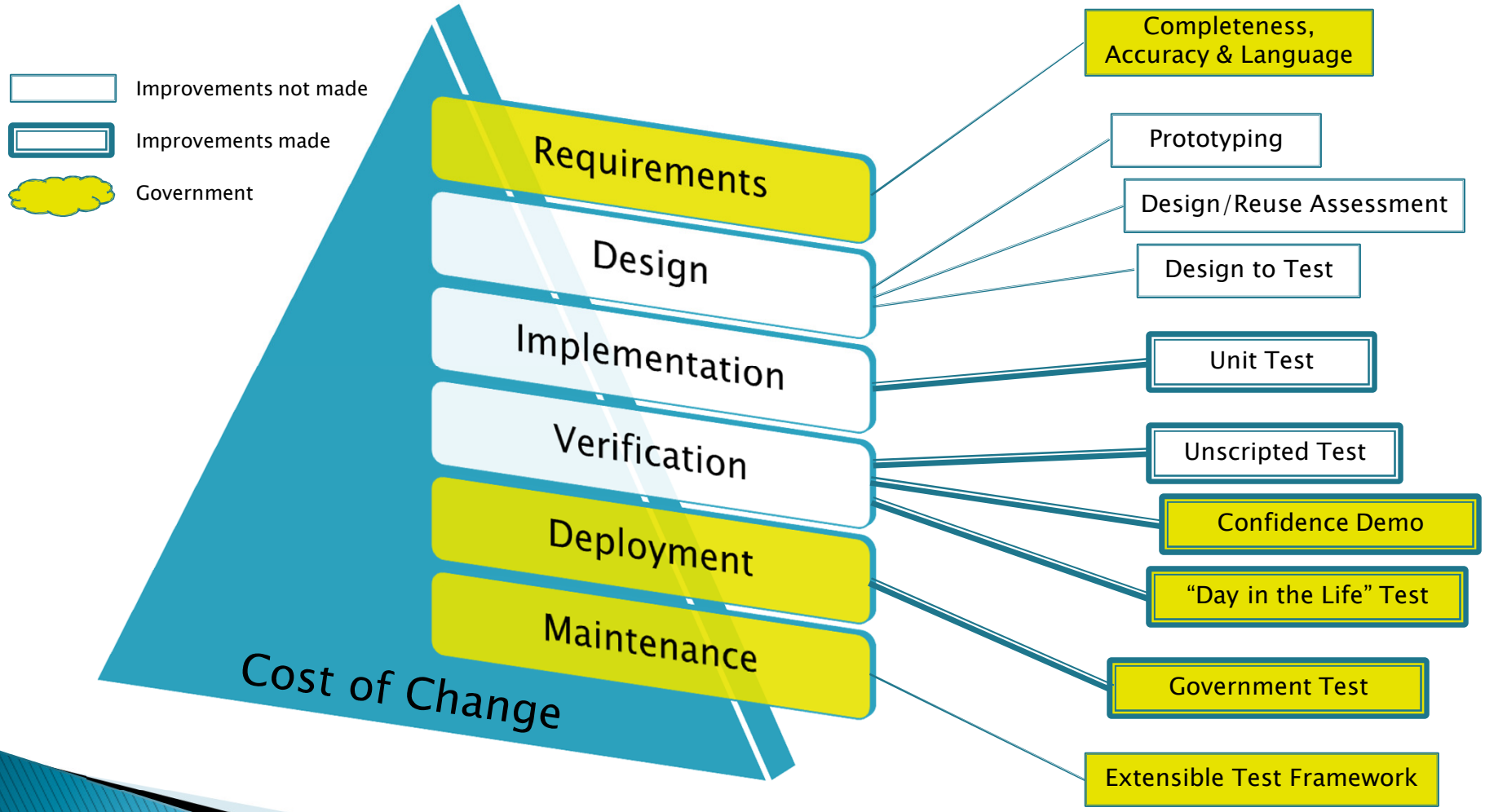


The Government's Role

- Develop Requirements
- Develop Request for Proposal
- Award Contract
- Oversee Execution
  - Manage Risk
  - Manage Contract Modifications
  - Manage Award Fee/Incentives
- Accept Product
- Oversee Sustainment

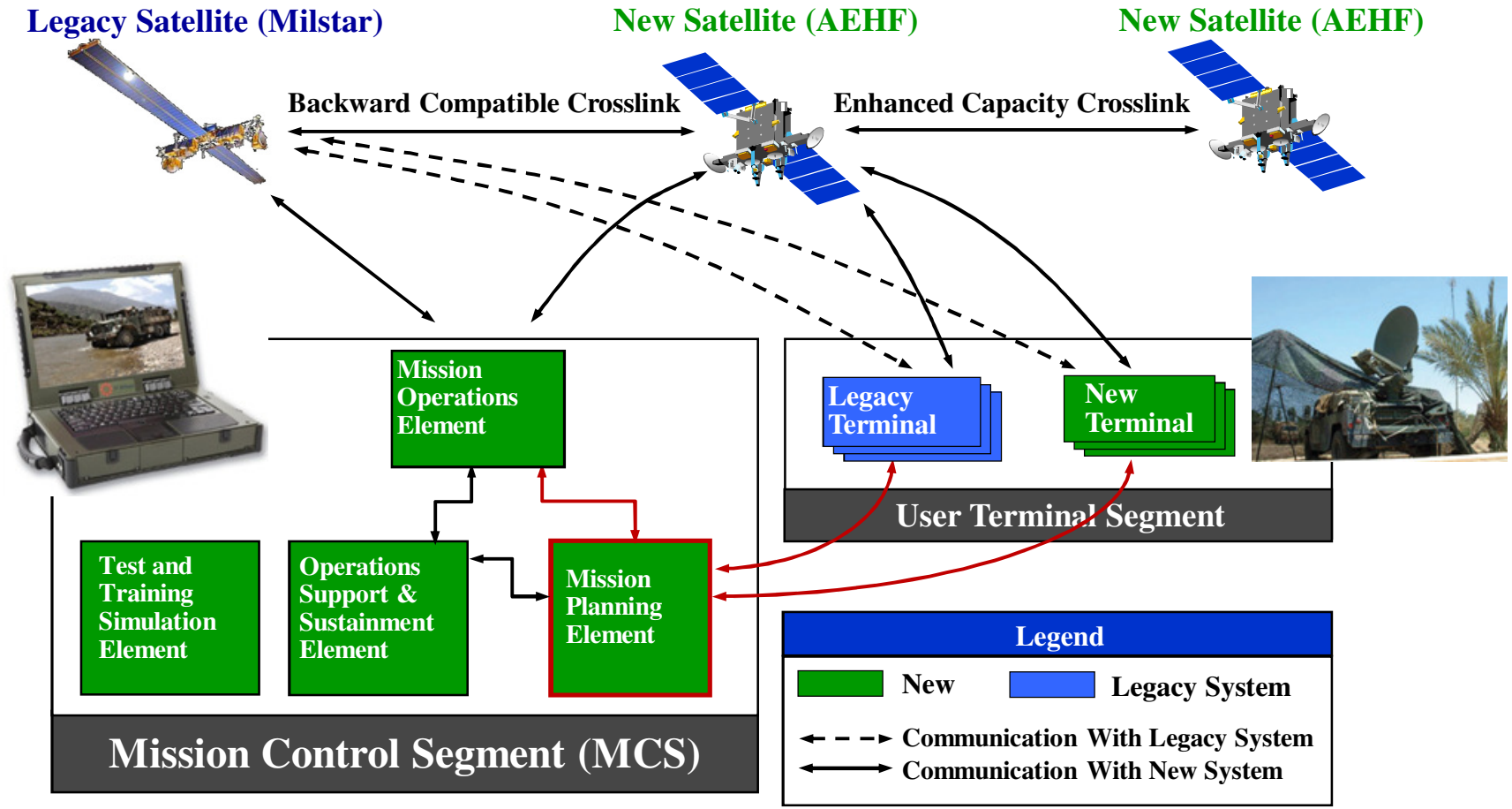
*Best Opportunities to Influence Outcome

The Pursuit of Quality/Suitability



System Architecture

Our System – Advanced EHF

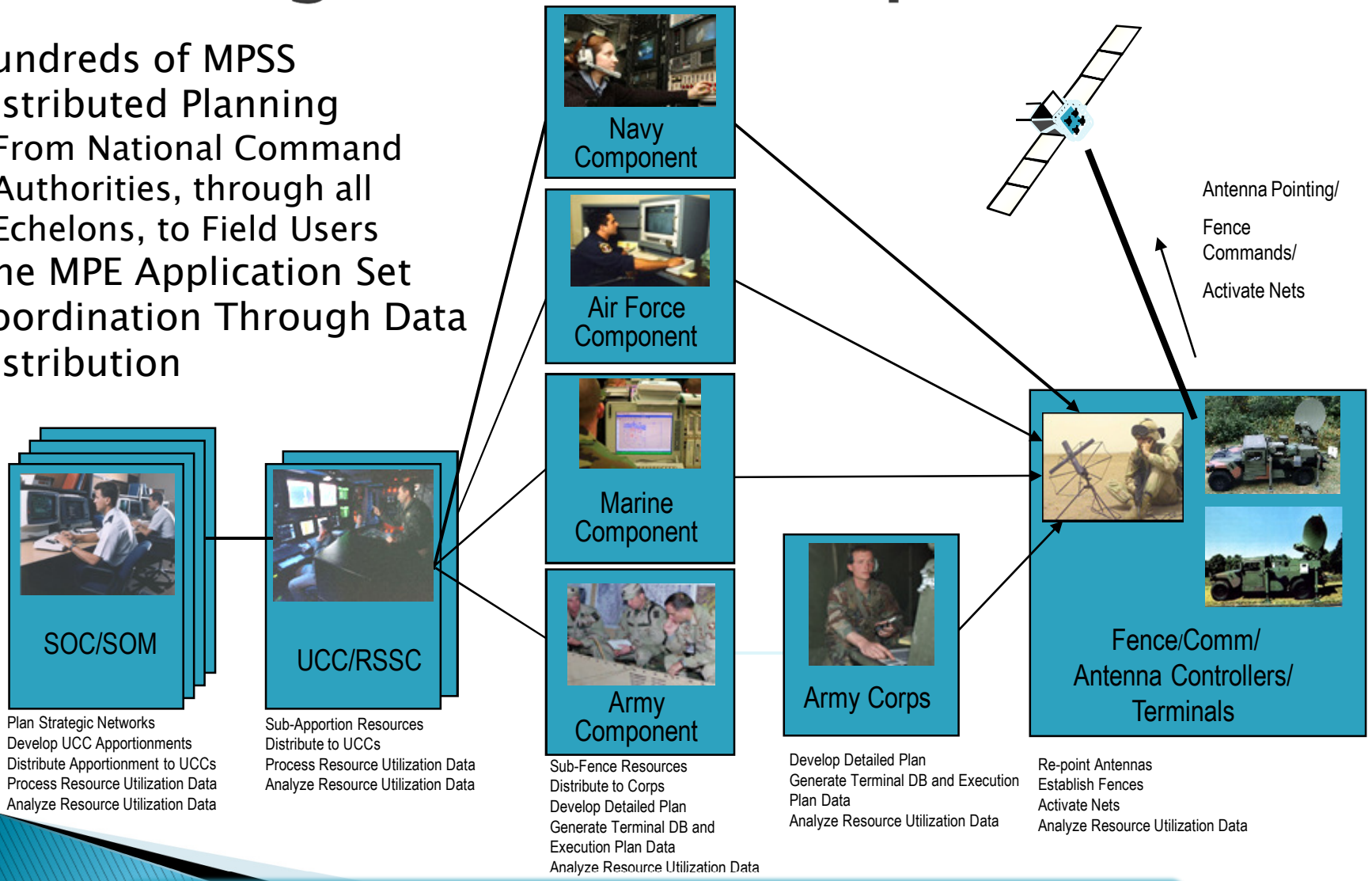


Mission Planning Element (MPE)

- Responsible for planning, scheduling, execution, data, and monitoring satellite and terminal resources
- Multiple users (100's) greatly varying in knowledge, skills and mission
- Extensive reuse of software from a previous system
- High complexity due to payload/terminal designs
  - 100's of screens, interdependent data
  - > 1.5M SLOC in C/C++
  - Software is classified SECRET

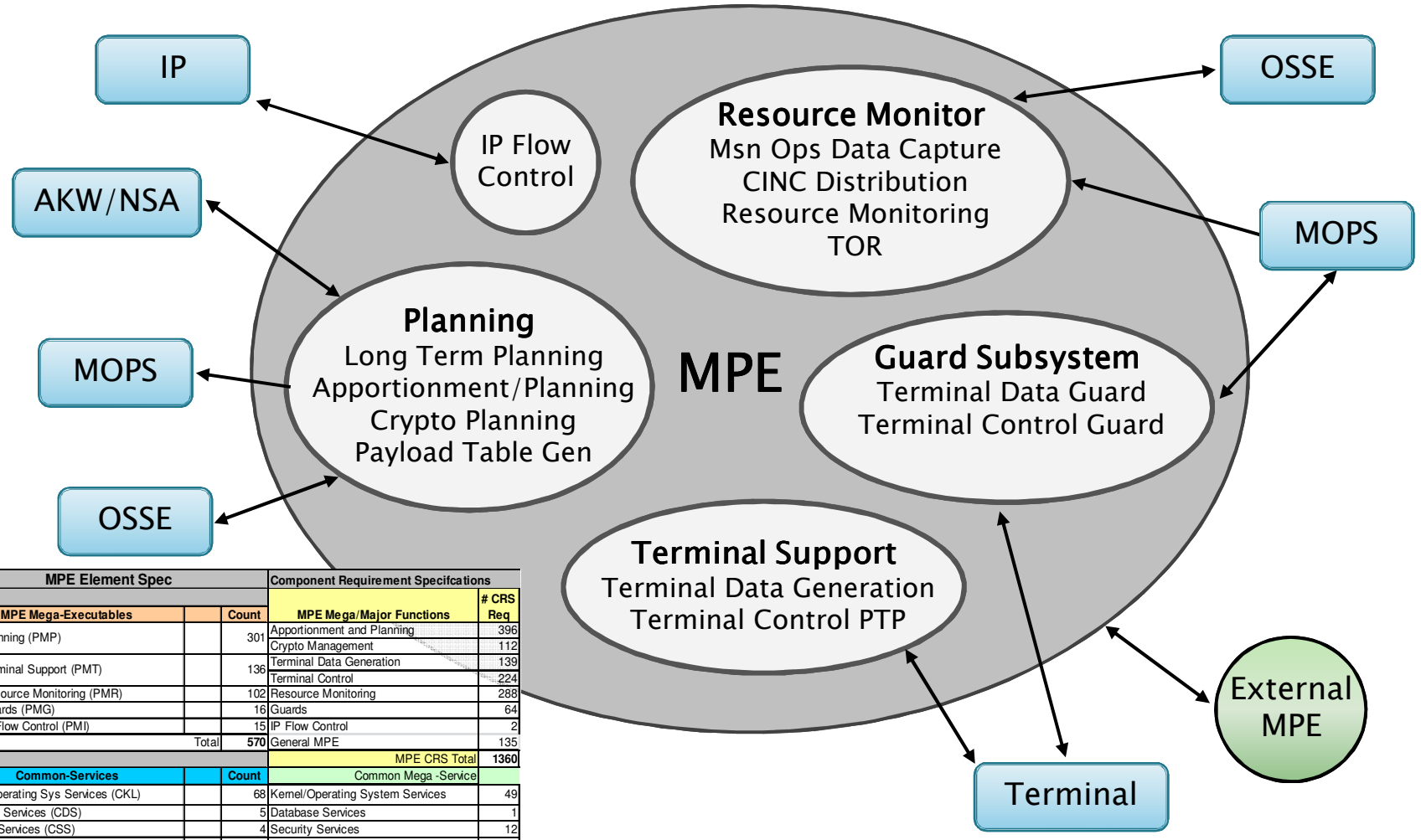
MPE High Level Conops

- Hundreds of MPSS
- Distributed Planning
  - From National Command Authorities, through all Echelons, to Field Users
- One MPE Application Set
- Coordination Through Data Distribution



Many Users / Many Different Missions

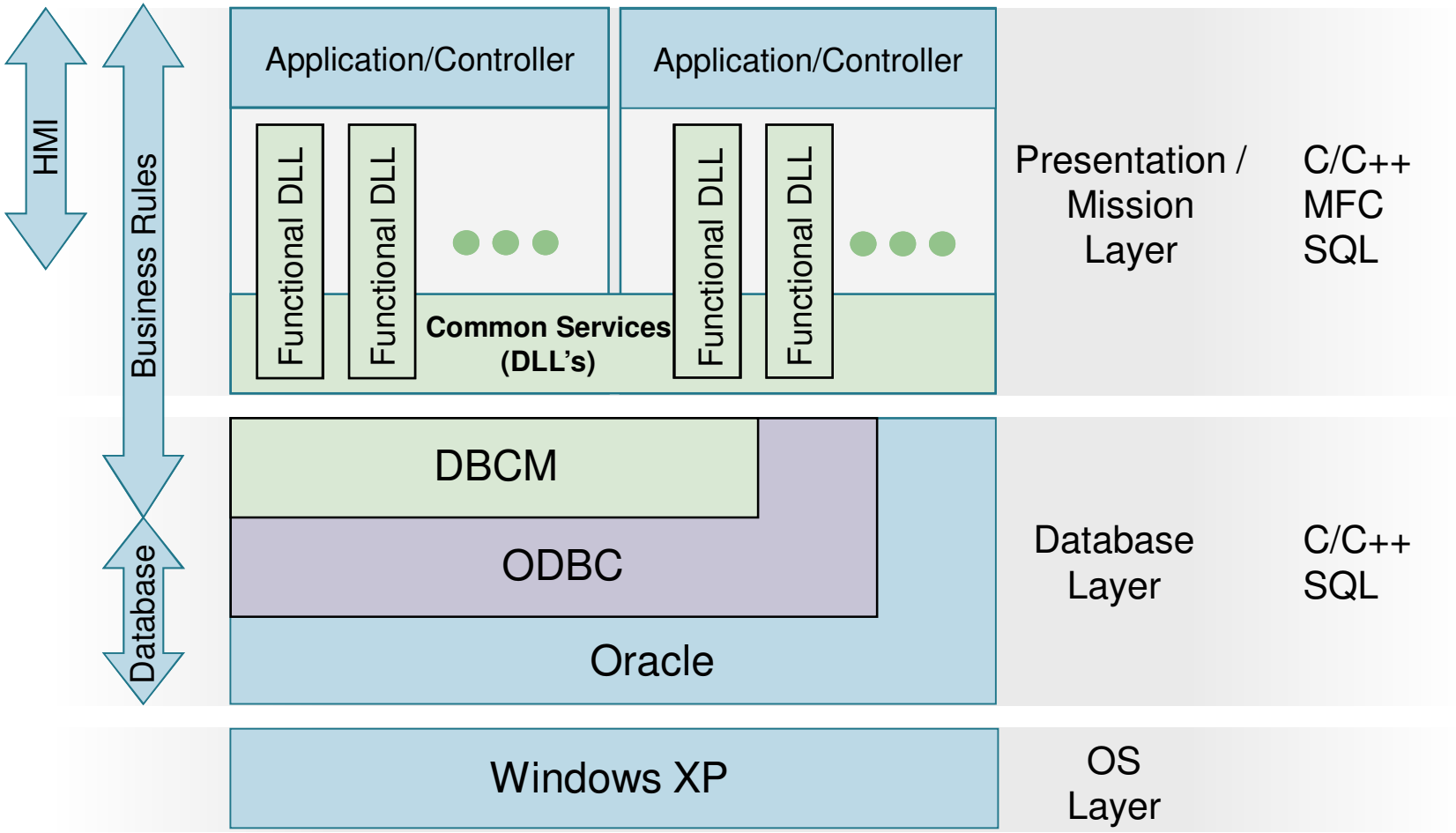
MPE Mission Executables



MPE Element Spec			Component Requirement Specifcations	
MPE Mega-Executables		Count	MPE Mega/Major Functions	# CRS Req
MPE Planning (PMP)		301	Apportionment and Planning	396
			Crypto Management	112
MPE Terminal Support (PMT)		136	Terminal Data Generation	139
			Terminal Control	224
MPE Resource Monitoring (PMR)		102	Resource Monitoring	288
MPE Guards (PMG)		16	Guards	64
MPE IP Flow Control (PMI)		15	IP Flow Control	2
	Total	570	General MPE	135
			MPE CRS Total	1360
Common-Services		Count	Common Mega -Service	
Kernel/Operating Sys Services (CKL)		68	Kernel/Operating System Services	49
Database Services (CDS)		5	Database Services	1
Security Services (CSS)		4	Security Services	12
Archive Services (CAS)		6	Archive	2
	Total	83	General COE	1
			Common Total	65

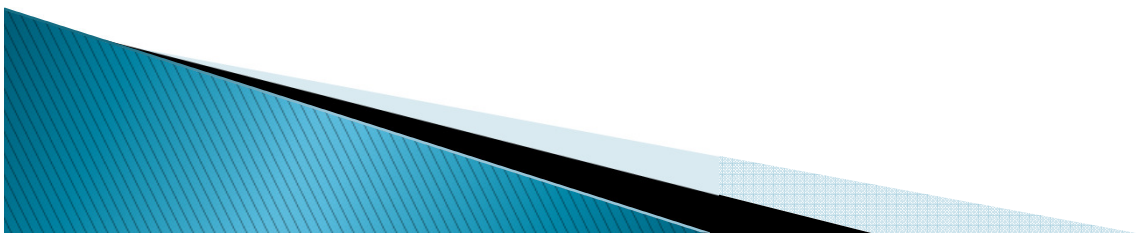
Many Requirements Spanning Multiple Executables

MPE Software Architecture



Business Rules Span Multiple Layers

How should you test a system like this?



Approach and Challenges

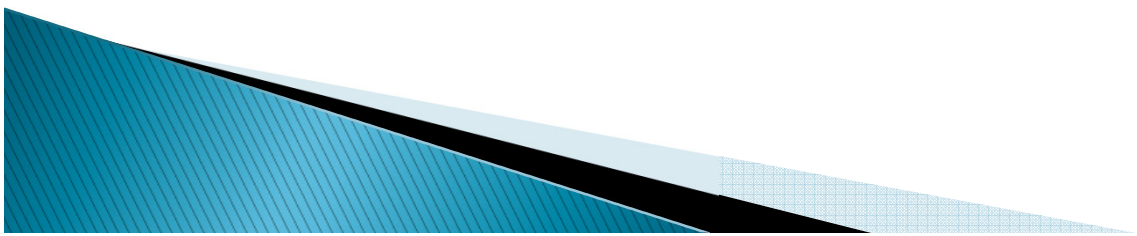
- It's Not the Same as Last Time

MPE Challenges

- Organizational Bias
  - Bias towards doing the way we did it last time
  - Desire to reuse previous standard test procedures
- Requirements Definition
  - MPE expected to be “all things for all users”
  - Requirements are never perfect
  - Evolving ConOps (and requirements interpretation)
- Requirements Verification
  - Requirements sold-off at lowest applicable level of testing
    - Unintended Regression Danger
  - Significant effort for testing of scenarios and DR removal remain after many requirements already “sold off”
- Schedule Pressure
  - Focus getting cases completed, not straying from the script
  - Testing this system difficult and time consuming

Scripted Testing Classic Problems

- Test Scripts Are Not Correct
  - Can't always capture the steps needed to prove the software was working correctly all the time
- Inattentional Blindness
  - Blind to errors that are not part of the test case
- Inaccurate Perceptions
  - Perception that completing test cases is, by itself, an accurate measure of progress – it is not
  - Perception that if you have passed the test cases the software is ready



Observations and Conclusions

- Overall
  - Product acceptance based on requirement verification alone not the correct strategy for a product as complex and flexible as MPE
- Specific
  - Formal tests typically use small input data sets
    - Testers stick to script, many operational logic paths left untested, lack of operational flavor in test
  - Insufficient system-level and operational environment experience among testers
    - Again, many operational logic paths not tested
  - Insufficient peer reviews
    - Defect escapes from design inspections, code inspections
  - Insufficient integration tests
    - Lack of diversity in testing

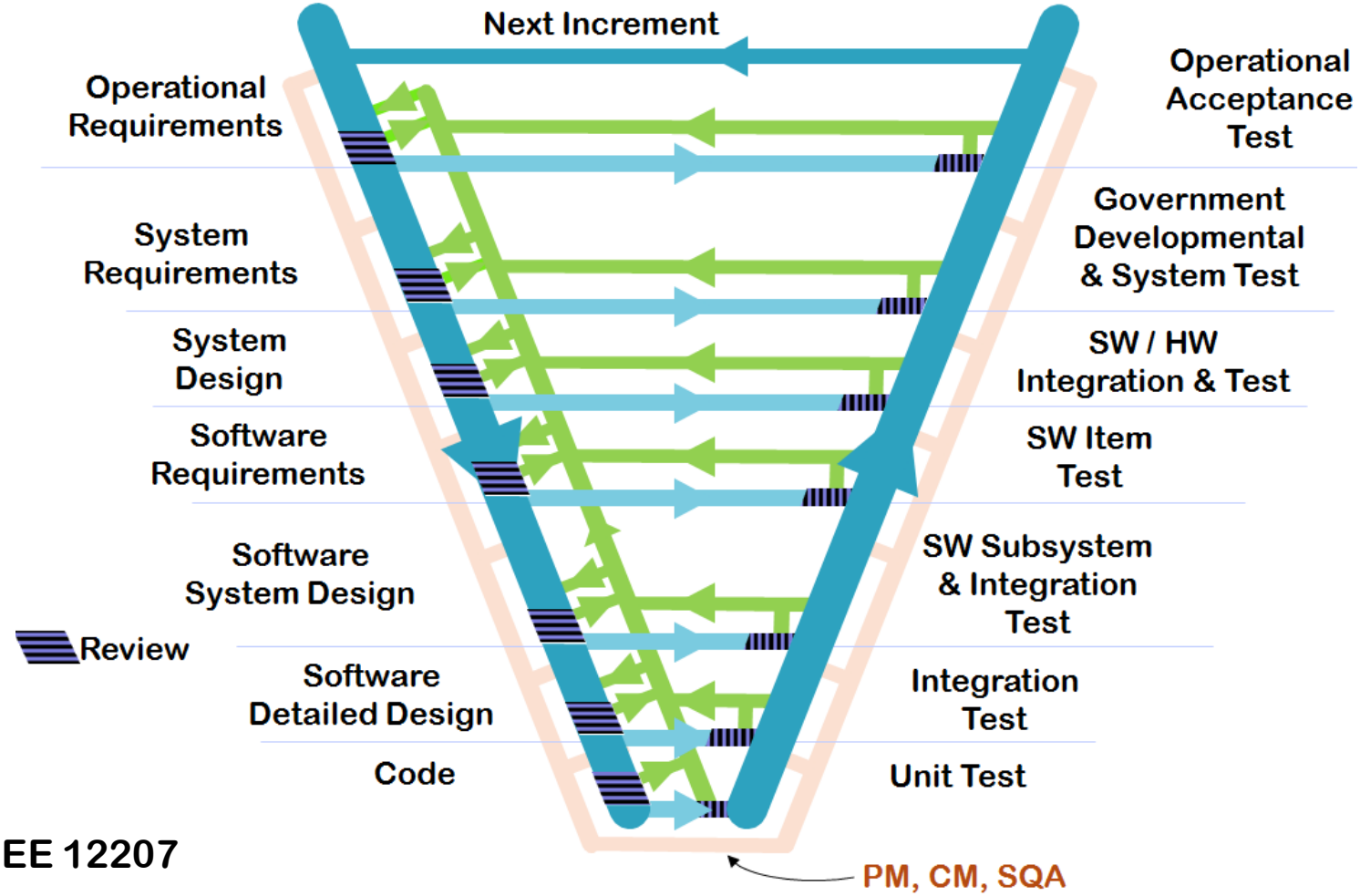
Test Process Improvements

Challenge	Corrective Action
Lack of operational oriented testing	• Added Independent Pre-Ops Test program
Peer reviews and integration and formal test approach insufficient	• Added Formal Integration Test (FIT) • Increased Peer Reviews
Requirements verification alone insufficient	• SPO acceptance after successfully completing government provided Day-in-the-Life (DITL) test

Unscripted Test

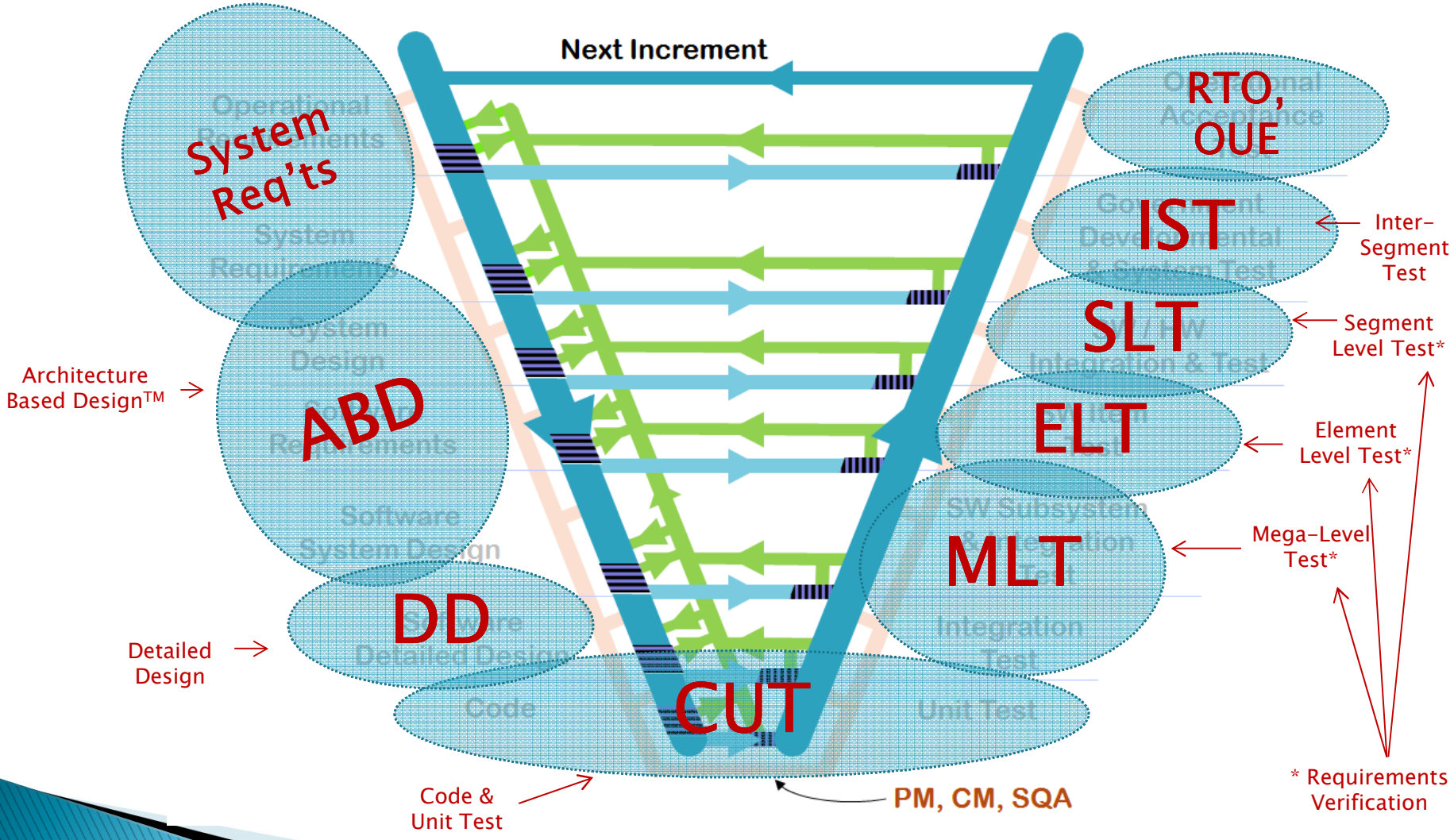
- Unscripted, not Unplanned

Classic Software Lifecycle Model

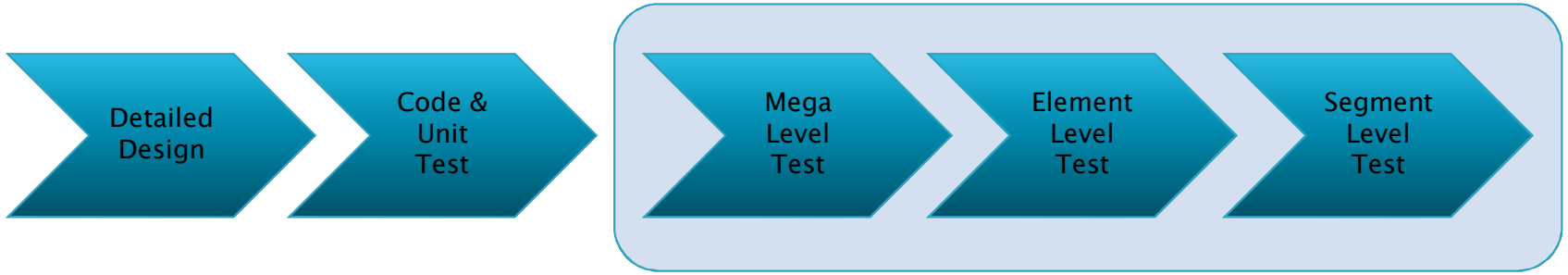


IEEE 12207

MPE Equivalent Lifecycle Model



Formal Test Program



Requirements Verification

↓

Test Cases Completed

↓

Earned Value Claimed

Unscripted Testing

- Form of quality checking that does not rely on test scripts. A tester is let loose on the system and encouraged to report all issues.
- How did we incorporate unscripted test in AEHF?
  - Formal Integration Test (FIT)
    - Contractor led mega-level integration test, freedom to test wide range of values
  - Pre-Ops Test
    - Independent test team focused on mission-based unscripted testing
  - Day-in-the-Life Test
    - Three day test written by government for contractor to assess product operability and effectiveness

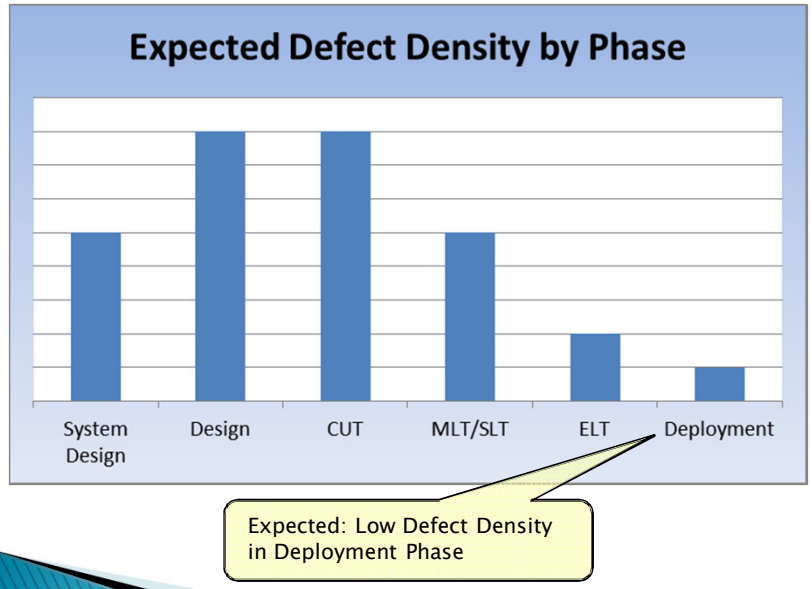
Formal Integration Test (FIT)

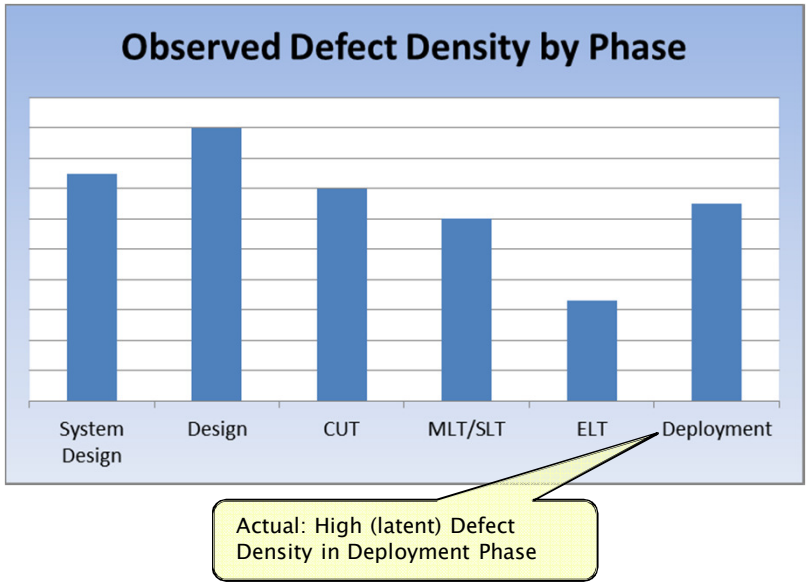
- Developer-led test
- Bridges gap between requirements testing and unscripted testing. Specifies fields to test, but freedom to test wide range of values.
- Matrices developed by examining fields and subset of variables to be tested
- Sequence of parameterized steps associated with a particular function. Variable of Interest x Range of Interest

Fields on Display	Range of Interest	Number of Items to Test	Total Number of Items Tested	Items Tested	Tester Initials	Deep Dive	Deep Dive section	Testing complete
Active Term Status Tab:		--	--					Yes
User Defined Filter	none	0	0					Yes
Start Date/Time	none	0	0					Yes
Stop Date/Time	none	0	0					Yes
Origin	none	0	0					Yes
Source Id	1, 2, term Id	3	1	1	TDG, SHT	RM-4, RM-3	4-3	No
Term Id	1 in US Range 1 in IP Range	2	1	2048 4000	TDG SHT	RM-4 RM-3	4-3	No
Term Type	1 L/M capable, 1 L/M/X capable, 1 XDR only capable	3	1	NMT SHORE SMART-T	TDG SHT	RM-4 RM-3	4-3	No
Owner	US parent org, US sub org	2	1	CENTCOM ARMY	TDG SHT	RM-4 RM-3	4-3	No
Event Time	none	0	0					Yes
Term Status Results Tab:		--	--					Yes
Event Time	none	0	0					Yes
Origin	none	0	0					Yes
Source Id	1, 2, term Id	3	1	1	TDG	RM-4	4-3	No
Term Id	1 in US Range 1 in IP Range	2	1	2048 4000	TDG SHT	RM-4 RM-3	4-3	No
Term Type	1 L/M capable, 1 L/M/X capable, 1 XDR only capable	3	1	NMT SHORE SMART-T	TDG SHT	RM-4 RM-3	4-3	No
Owner	US parent org, US sub org	2	1	CENTCOM SMART-T	TDG SHT	RM-4 RM-3	4-3	No
Sat Id	any 1 M* sat, any 1 AEHF	2	1	3 9	TDG SHT	RM-4 RM-3	4-3	No

Why Formal Integration Testing?

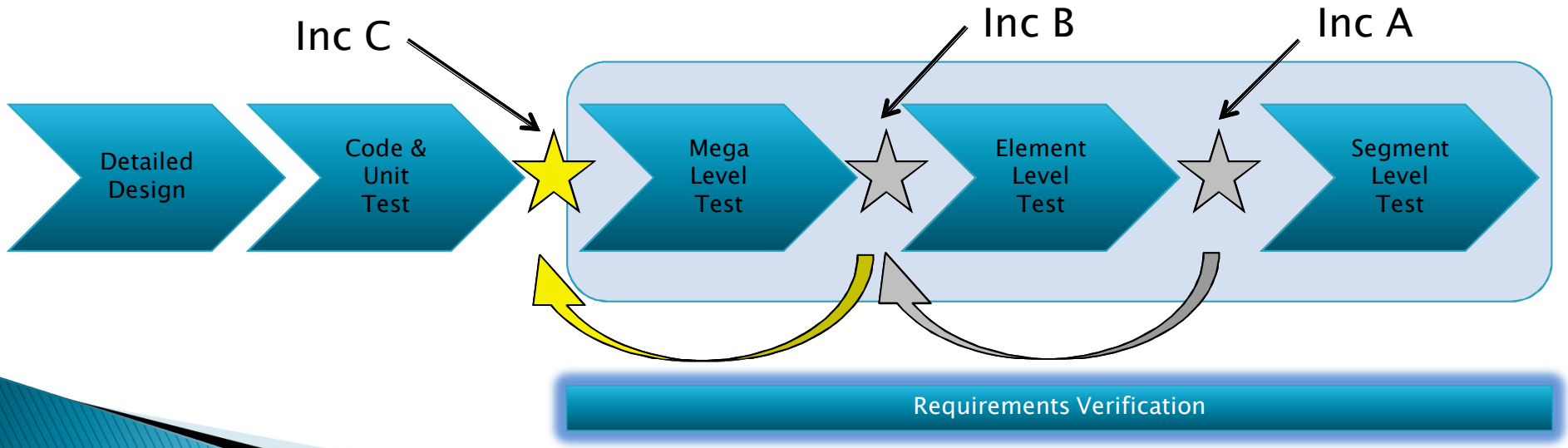
- Product experienced latent defect discovery in deployment and acceptance testing
- Process analysis highlighted need to enhance procedure driven requirement-based testing with scripted free-form combinatorial testing



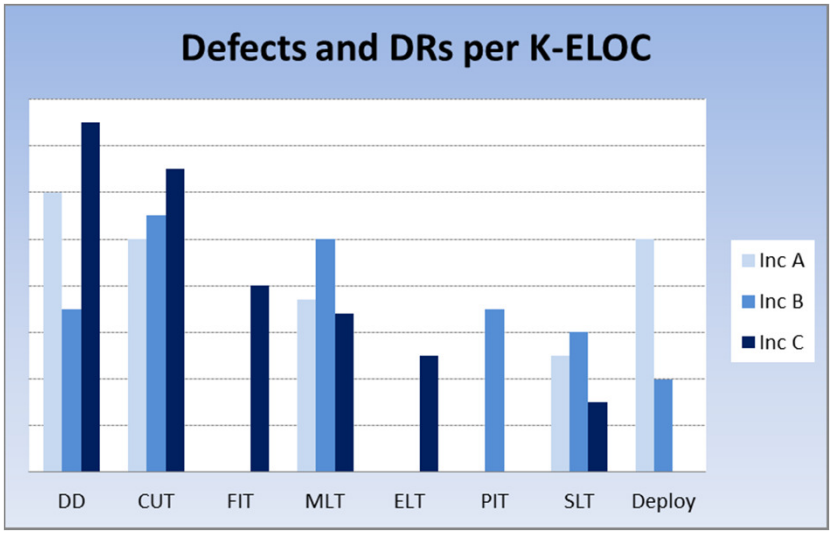


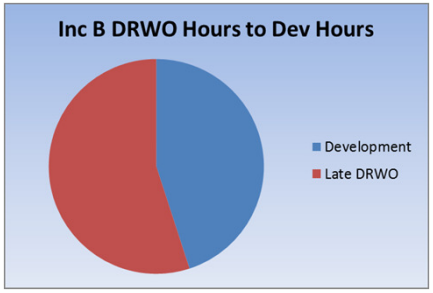
Finding the Right fit for FIT

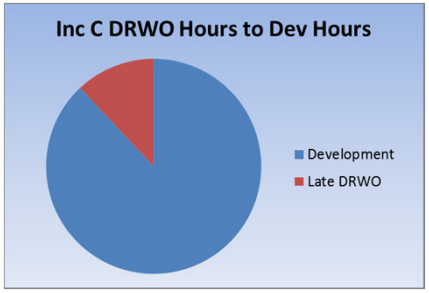
- Formal Integration Testing (FIT) was used across three increments
- The benefit of FIT increased as it moved left, driving out problems earlier in the development lifecycle

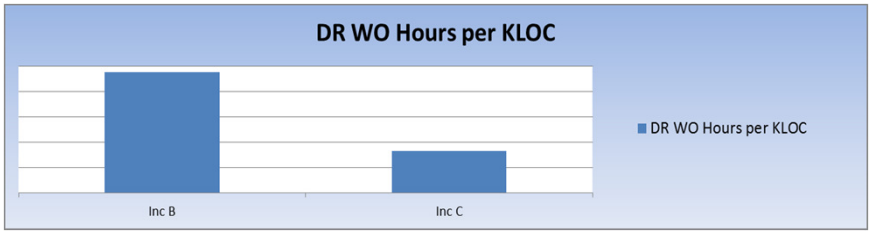


Value of Formal Integration Testing





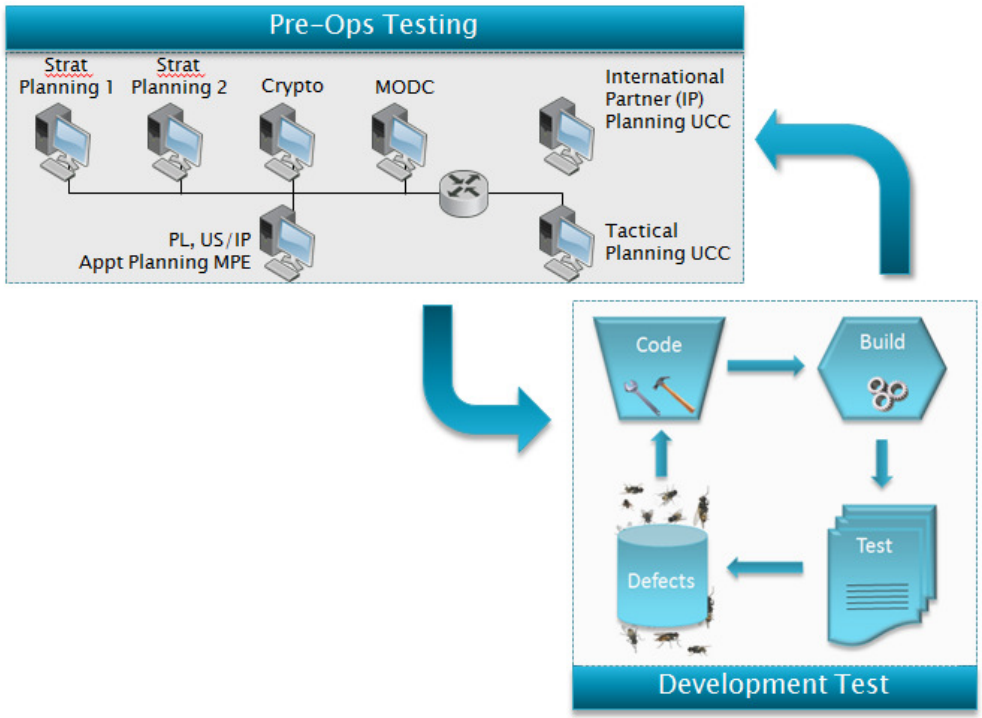




- Reduction in cost: Industry research shows that problems are easier (and cheaper) to fix earlier in the lifecycle
- Reduction in schedule: Fewer problems = fewer test re-starts = shorter test program duration
- Increase in confidence: Higher quality product delivered with fewer defects and fewer required workarounds

What is Pre-Ops Testing?

- Independent test team (non-developers)
- Mission-based unscripted testing
- Mission/domain experts
- Resembles a Beta Program



Hand-Holding vs. Pointing in Right Direction

Test Engineer

Formal Test Case

Create a communications plan and add the set of service configurations, BSMRCA Events, and Beam Events to that communications plan.

- On the BSMRCA Events Tab, add the following Event to the Event List:
  - Sat: 10
  - Beam: BSMRCA-1
  - Duty Cycle: 75, 25
  - Date/Time: Start time is default, start of Communications Events.

Motivation: Complete test cases for verification on or before schedule

VS.

Functional Experts

Pre-Ops Test Objective

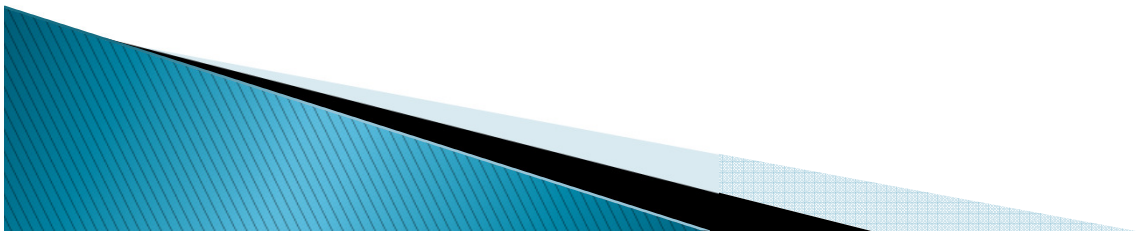
Plan for a total force deployment (100 terminals, 40 communication services)

- Plan for a total force deployment (100 terminals, 40 communication services) in a common Comm Plan using all Service types (combination of LDR/MDR/XDR to include OTADD & Navy RB Nets)

Motivation: Find as many defects as possible (within schedule)

Value of Pre-Ops Testing

- Mission Ops-Based Focus
  - Ops-like users performing real-world mission tasks
  - Accounts for multiple missions, users and levels of experience and knowledge
- Similar to typical “beta” testing
  - Users using product in unanticipated ways
  - Delivers more diversity in testing
  - Uncovers “user experience” issues
- Finds defects in development, not in operations



Government Developed Day-in-the-Life Testing

- Put Your Money Where Your Mouth Is

Issue

- Requirements verification alone inadequate for proving operability and effectiveness of the MPE product
  - Lack of diversity in testing
  - Users finding defects that should have been found earlier
  - Users finding issues in real-world conditions not represented in factory test environment

DITL Objective

- The primary objective of the DITL was to assess the operability and effectiveness of the MPE product
  - Can it accomplish a real-world mission under real-world conditions?
  - Is it ready for the user?

DITL Test Case Description

- Excerpt from Segment Level Test Plan

1.1.1.1 (U) Test Case I7-SLT-S007-003

(U) Days-in-the-Life MPE Planning

1.1.1.1.1 (U) Test Case Description

(U) The objective of the Increment 7 MPE DITL is to assess operability and effectiveness of the increment 7 MPE product. It will be run once all other Increment 7 requirements and product integration testing are completed. The test will be operationally relevant and exercise a broad array of MPSS capabilities consistent with the Increment 7 functional baseline. **There will be no traditional test script driving the activities at the defined MPSS stations. Instead, MPSS operators execute a set of Government provided tasks nominally expected to occur in operations.** These tasks may span multiple MPSS User Roles and may or may not need to occur sequentially. Each task will be sized appropriately with respect to nominal operational processing and the total effort of all tasks is not expected to exceed 72 hours of test conduct.

- The Contractor will request the MPE DITL test tasks after completing all predecessor test cases to the MPE DITL and baselining all MPSS DITL impacting DRs.

(U) There are no requirements that will be verified in this test case

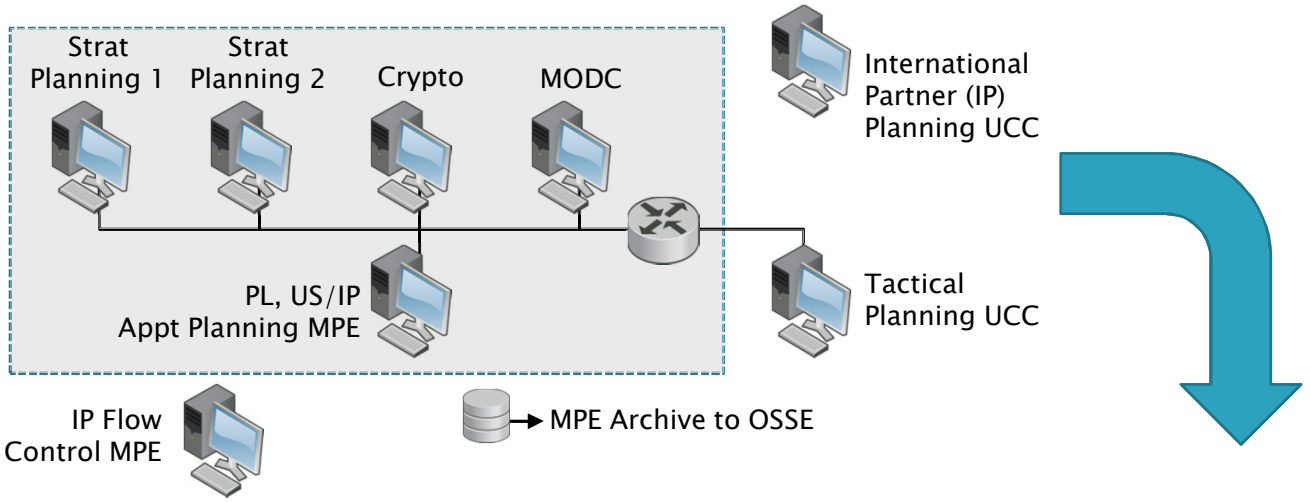
DITL Characteristics

- Operational Configuration
- Operational Inputs/Outputs
- Contractor personnel as users
- Mission Driven/Mission Scenario
- Contractor does not know ahead of time what will be on the test
- Passing DITL is an exit criteria for Segment Level Test
- Must be able to accomplish the mission
- Operationally acceptable workarounds OK

DITL Timeline

Timeline	Event
- 90 Days	Baseline TEM. Agree on starting DB and content, unique test configuration requests
- 30 Days	Walk Through Event. Multi-Day Site Walk-through of anticipated procedures, test threads, data capture methods, etc.
- 2 Weeks	Contractor Indicates "Ready To Execute"
- 3 Days	Contractor Requests MPE DITL Test Scenario and Government confirms entry criteria met (Successful TRR)
- 3 Days	Government provides DITL Test scenario and supporting data. The government and a contractor representative will review the mission planning scenario to ensure the missions are understood, in scope and relevant to the test objectives.
+ 0 Days	Execution of DITL scenario steps begins. (End of day tag-ups: (1) Contractor, (2) User Representatives)
+ 8 Days	Execution completes after 72 hours of test execution time or successful completion of steps
+ 10 Days	Complete Analysis and Government Chief Software Engineer issues findings (Pass/Fail)

DITL Operational Configuration



	USER ROLES							
	USER ROLES	Strat1	Crypto	PL,US/IP Appt Planner	Strat2	Crypto	Tactical Planner	IP Planner & DCA *	4SOPS RM
	Hardware	MPE1	MPE2	MPE3	MPE4	MPE2	MPE5	MPE6 IP Flow*	MODC1 MPE3
Mission Threads	Day 1	Migration							
	Day 2	Strategic Planning And Operations			Nominal Planning				Nominal Resource Montoring
	Day 3				Implementation And Distribution			IP Planning	

DITL Input Scenario Example

Task Description

Input Data in Operational Format

Evaluation

Created Output Products for Later Evaluation

Strategic Planning						
Task ID	Scenario	Name	Data	Eval Criteria	Output Product to PDF and save to directory (e.g. I:\DitLProducts\MMYYYY\	Time Est.
25.	IMAGE	In Endurance Mission Re-plan, fail M1	Take screenshots of the Impacted Services/Terminals HMI	Re-planned terminals and services following criteria in requirement MPE2805.	Comm Plan Summary Report, Terminal and Service Configuration Reports for "Impacted Services/Terminals"	1 hr
26.	IMAGE	The following tasks will be performed to verify functions independently, but are not dependent on previous tasks for strategic planning				
27.	IMAGE	Move M4 sat in prep for A3	Reference Tab S-8	Sat moved properly	N/A	10 min
28.	IMAGE	Add A3 in Late-Transition mode	Reference Tab S-9	Sat created & payload(s) configured properly	Payload Configuration reports	20 min
29.	IMAGE	Configure crosslinks	Reference Tab S-9	All sats cross-linked successfully	LDR/MDR/XDR Constellation Configuration Reports	15 min
30.	IMAGE	Generate A3 Payload Table (using Payload Table Generation Application)	AEHF DCID 51 (Downlink Channel Parameters Table)	P/L Table file generated	N/A	5 min
31.	IMAGE	Make A3 changes in Frequency Planning Wizard	Reference Tab S-10	Frequency Planning Wizard completes	Screenshots of Demod Map and WCB Map	20 min
32.	IMAGE	Save changes to A3 frequency plan as "Late Transition Modified" template	N/A	Template shows in Frequency Planning Template list	XDR Frequency Planning Report	10 min

Table 1 – Strategic Planning Tasks (9.5 hours)

Value of DITL

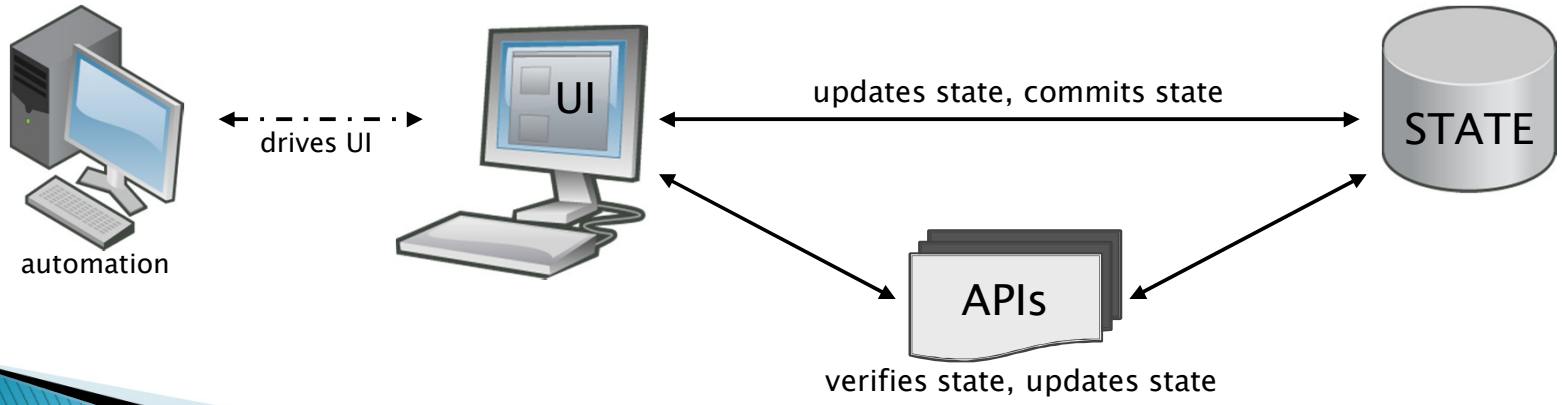
- Provided a test gate based on proving the software could accomplish a real-world mission
- Gave the government leverage to assess suitability for ops despite no formal suitability requirement
- Incentivized the contractor to introduce new forms of mission based testing using real-world operators (Pre-Ops Testing) to ensure they could get through this gate
- Dramatically improved the quality of the product

Incremental Test

- Progress, not Regress

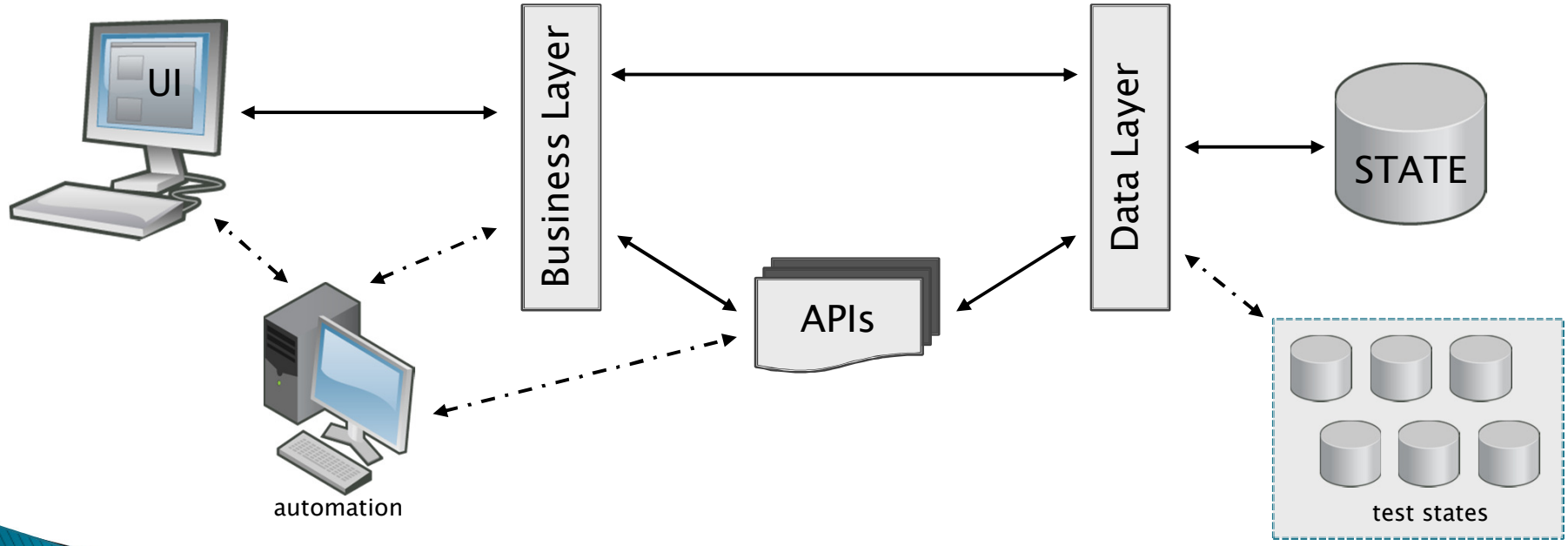
Automate, Automate, Automate

- A system that is ridiculously hard to test is equally ridiculously hard to regression test
- Automated testing expedites verification and protects against regression
- Some systems are more difficult to automate than others

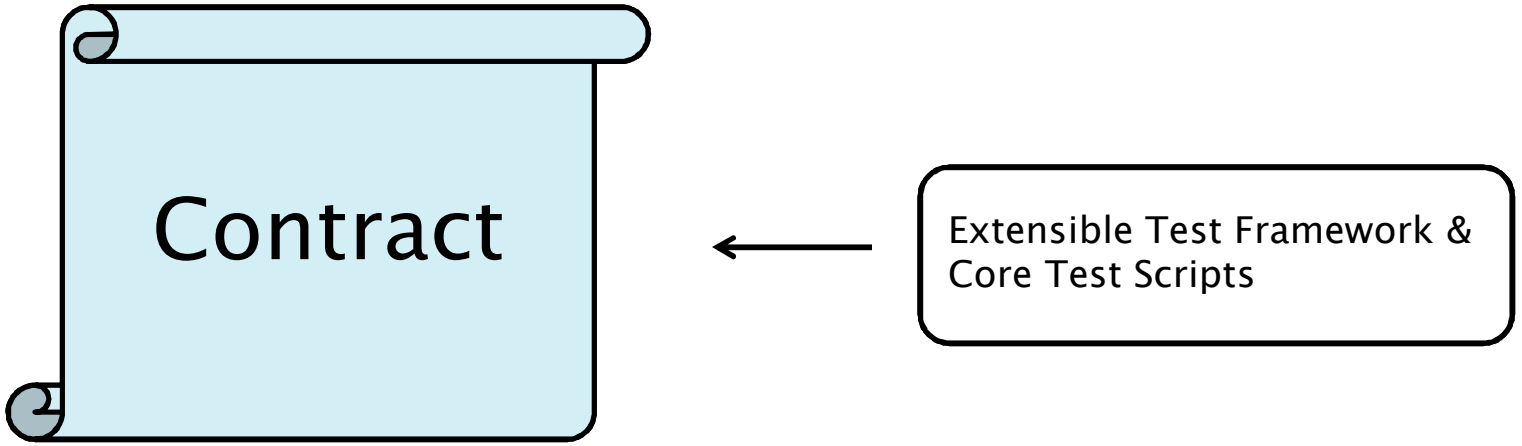


Design for Test

- Implementing automated testing can be simplified/facilitated through design

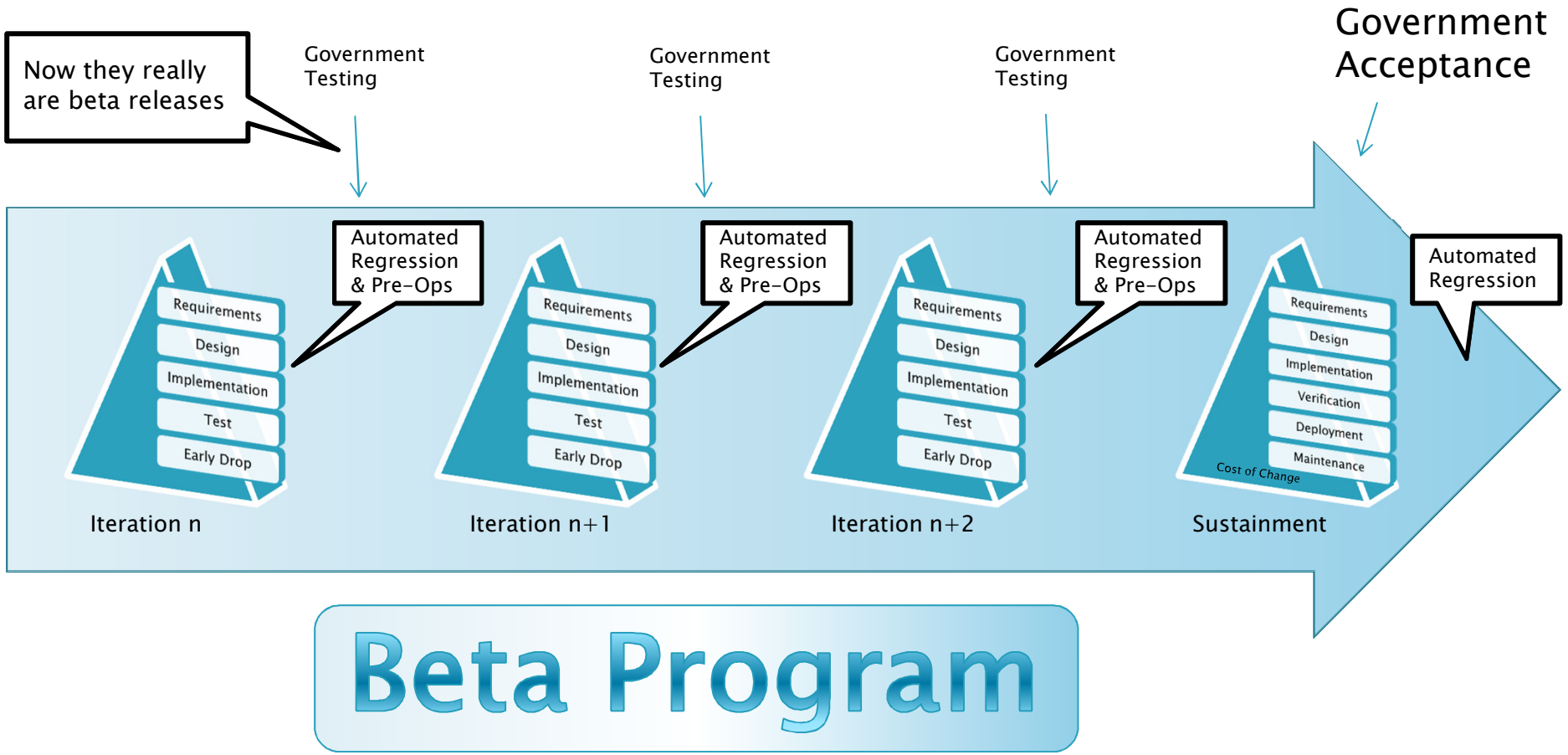


Deliverable Test Scripts



- Protects against regression
- Improves and expedites verification
- Promotes good architecture and design
- Supports sustainment activities

From Increments to Early Drops



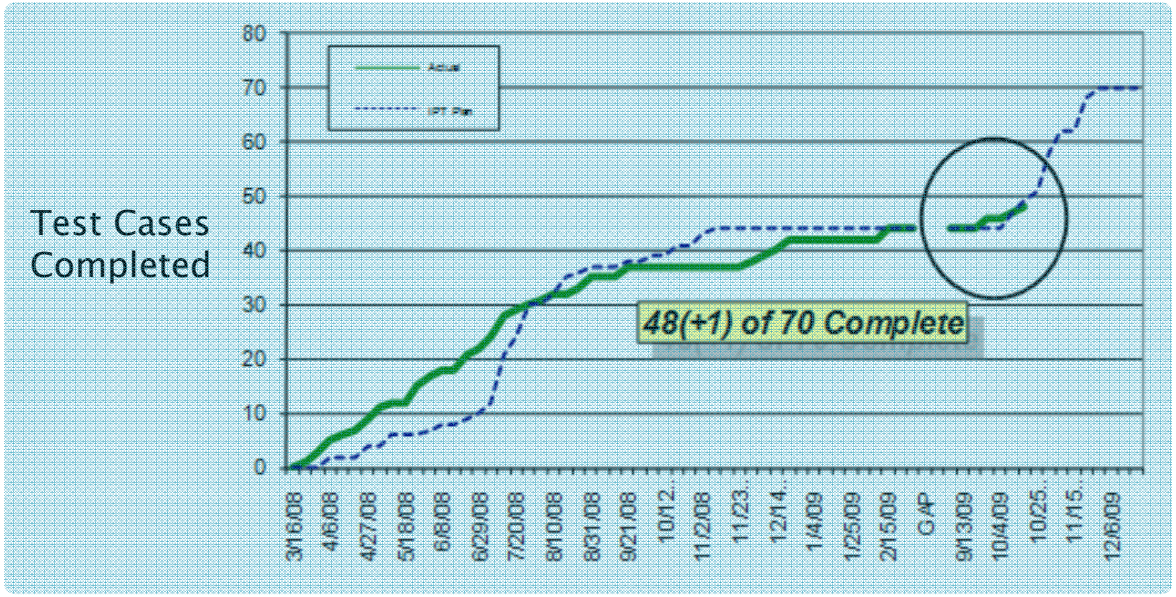
Beta Program

Risk Management

- Quality vs. Progress

Measuring Progress vs. Quality

- Progress or Quality?



- Completed test cases may not reflect quality
- Earned Value claimed on “defects closed” rather than “defects remaining open” can be a misleading indicator of progress

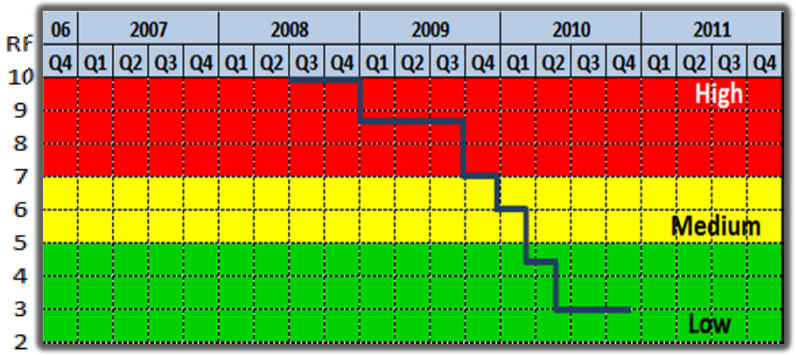
Software Maturity Risk

Initial Risk Definition

- **If the software products have a significant quantity of DRs (including DRs of a high severity) and/or a large quantity of projected latent DRs, then the ability to execute System Test will be adversely impacted, program schedules could be delayed and the ability to field the software could be adversely impacted.**

- **Risk Areas:**
  - **Unplanned integration problems / rework**

Burn Down Schedule

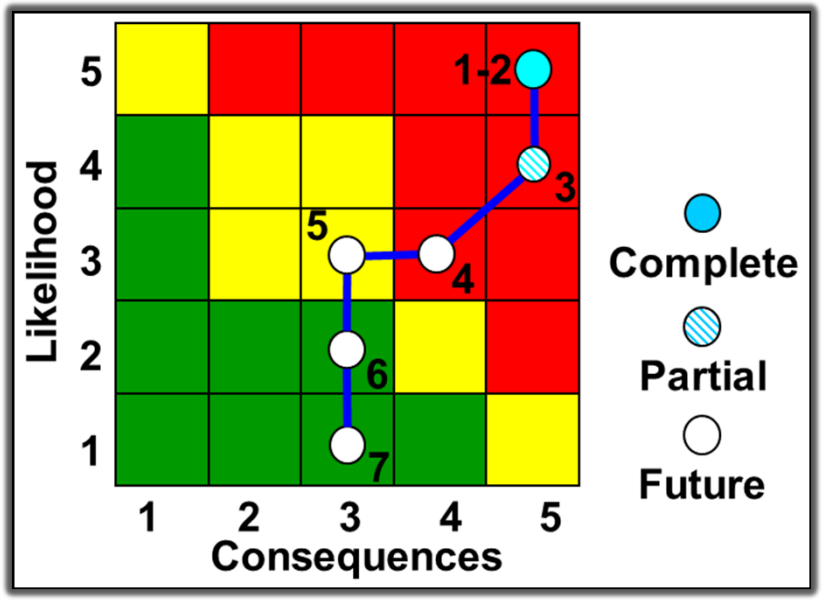


Mitigation Plans

	Milestone	Risk Level
1	Complete DR Assessment	R 5L,5C
2	Weak SW Reliability Detected	R 5L,5C
3	SW Reliability Growth Confirmed	R 4L,5C
4	Strong SW Reliability Growth Detected	R 3L,4C
5	Strong SW Reliability Growth Confirmed	Y 3L,3C
6	Day-In-The-Life (DITL) Completion	G 2L,3C
7	Monitor Inc 5 Maturity Through IST (6K-2)	G 1L,3C

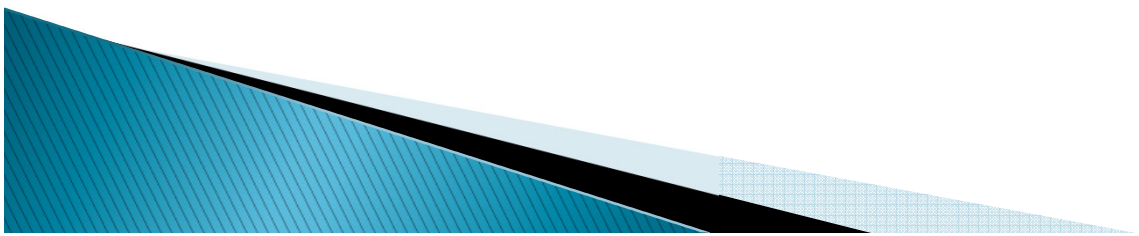
Challenge to measure SW Reliability – hrs/defect, defect density, defect priority, defect backlog

DitL is the final milestone for Green



Managing Risk

- An unscripted, government developed test like the DitL can improve software quality
  - Establishes a mentality with the contractor that the software just has to work!
  - Less about the metrics, more about functionality and suitability
- How do we measure software maturity?
  - Common understanding of how defects are captured/measured
    - Normalized metrics, Intuitive front-end for viewing defects
  - Assuming a constant level of test and fix rate, defect backlog should be decreasing
    - Completing test cases is not a measure of quality
  - Automated regression



The Contract and Award Fee Plan

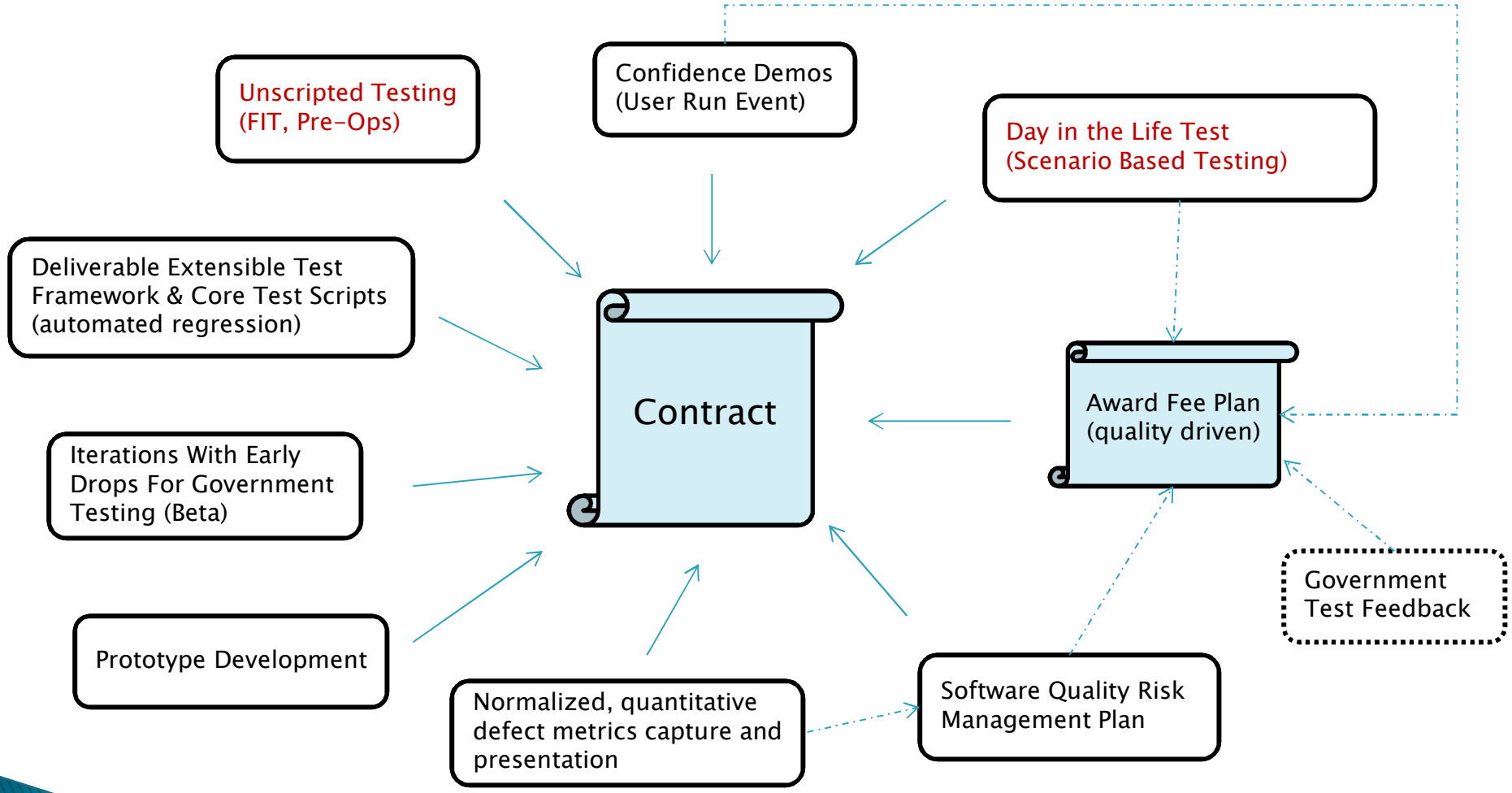
- Where Rubber Meets Road

You Get What You Ask For

- The contract and award fee plan set the stage for program execution and, ultimately, the quality of the product delivered

- The following performance criteria encourage schedule over quality
  - Technical Milestone Performance
    - Deliver a product that meets requirements
  - Cost Performance
    - Earned Value Management (maintain good SPI & CPI)

The Contract



Final Thoughts

- Deploy Quality


"TEST LIKE YOU FLY"

Summary: Focus On Quality and Suitability

- Bake unscripted testing into the contract
- Include government developed tests as quality gates, eg. DitL
- Consider including deliverable automated test scripts as a part of the contract
- Scrutinize proposed test strategy and make sure that it is appropriate for the system being developed (requirements verification is seldom enough with today's software)
- Risk Management Plan should provide a quantitative way to measure software quality as a positive trend and a measurable mitigation plan to address deficiencies
- Award the contractor for delivering quality products. Avoid "binary" criteria that encourage schedule over quality
  - NOT GOOD ENOUGH: Deliver a product that meets requirements
  - BETTER: Deliver a product that meets requirements **AND** exhibits a high level of quality and suitability as qualified by:
    1. Risk mitigation activities (are we effectively identifying and burning down risk?)
    2. Quantitative metrics (DDPs, Hrs/Defect, Real-time Dashboard Views) for trends, not just threshold points
    3. User input from government test and confidence demos
- Do whatever you can to get the user community involved early – design reviews, prototypes, confidence demos and early drops for government testing
- Consider iterations and early drops over formal incremental deliveries. Making something operational is expensive and time consuming
  - You might (will) end up with costly overruns if (when) the schedule slips

Thank You

Acronym List

- AEHF – Advanced Extremely High Frequency
- CM –Configuration Management
- CPI – Cost Performance Index
- DBCM – Database Consistency Manager
- DDP – Defect Density Profile
- DITL – Day in the Life
- DLL – Dynamic Link Library
- FIT – Formal Integration Test
- HMI – Human Machine Interface
- HW – Hardware
- IP – International Partner
- LDR – Low Data Rate
- MDR – Medium Data Rate
- MCS – Mission Control Segment
- MFC – Microsoft Foundation Classes
- MODC – Mission Ops Data Capture
- MOPS – Mission Operations Element
- MPE – Mission Planning Element
- MPSS – Mission Planning Sub-System
- ODBC – Open Database Connectivity
- OSSE – Operations Segment Sustainment Element
- OUE – Operational Utility Event
- PM – Program Management
- RSSC– Regional SATCOM Support Center
- RTO – Responsible Test Organization
- SOC – Satellite Operations Center
- SOM – System Operational Manager
- SPI – Schedule Performance Index
- SQA – Software Quality Assurance
- SQL – Structured Query Language
- SW – Software
- UCC – Unified Combat Control
- XDR – eXtended Data Rate